Assignment No:6

Title:	Personnel information system using sorting and searching for STL and vector container.
Problem Statement:	Write C++ program using STL for sorting and searching user defined records such as personal records (Name, DOB, Telephone number etc) using vector container. **OR** Write C++ program using STL for sorting and searching user defined records such as Item records (Item code, name, cost, quantity etc) using vector container.
Prerequisites:	
Object Oriented Programming	
Objectives:	
To learn the concept STL, searching, sorting and vector container.	
Theory:	

STL:

The Standard Template Library (STL) is a set of C++ template classes to provide common programming data structures and functions such as lists, stacks, arrays, etc. It is a library of container classes, algorithms, and iterators. It is a generalized library and so, its components are parameterized. A working knowledge of template classes is a prerequisite for working with STL.

STL has four components

- Algorithms
- Containers
- Functions
- Iterators

Algorithms

- The algorithm defines a collection of functions especially designed to be used on ranges of elements.They act on containers and provide means for various operations for the contents of the containers.
- Algorithm
  - Sorting
  - Searching
  - Important STL Algorithms
  - Useful Array algorithms
  - Partition Operations
  - Numeric

Containers

- Containers or container classes store objects and data. There are in total seven standard "first-class" container classes and three container adaptor classes and only seven header files that provide access to these containers or container adaptors.

- Sequence Containers: implement data structures which can be accessed in a sequential manner.

- vector
- list
- deque
- arrays
- forward_list(Introduced in C++11)

- Container Adaptors : provide a different interface for sequential containers.
  - queue
  - priority_queue
  - stack
- Associative Containers : implement sorted data structures that can be quickly searched (O(log n) complexity).
  - set
  - multiset
  - map
  - multimap
- Unordered Associative Containers : implement unordered data structures that can be quickly searched
  - unordered_set
  - unordered_multiset
  - unordered_map
  - unordered_multimap

Functions

- The STL includes classes that overload the function call operator. Instances of such classes are called function objects or functors. Functors allow the working of the associated function to be customized with the help of parameters to be passed.

Iterators

- As the name suggests, iterators are used for working upon a sequence of values. They are the major feature that allow generality in STL.

Utility Library

- Defined in header <utility>.
- pair

Sorting:

It is one of the most basic functions applied to data. It means arranging the data in a particular fashion, which can be increasing or decreasing. There is a builtin function in C++ STL by the name of sort(). This function internally uses IntroSort. In more details it is implemented using hybrid of QuickSort,
HeapSort and InsertionSort.By default, it uses QuickSort but if QuickSort is doing unfair partitioning and taking more than N*logN time, it switches to HeapSort and when the array size becomes really small, it switches to InsertionSort. The prototype for sort is :

sort(startaddress, endaddress)
startaddress: the address of the first element of the array
endaddress: the address of the next contiguous location of the last element of the array. So actually sort() sorts in the range of [startaddress,endaddress)

```
//Sorting
    #include <iostream>
    #include <algorithm>
     usingnamespacestd;
```

```
voidshow(inta[])
{
   for(inti = 0; i < 10; ++i)
      cout<< a[i] << " ";
}

intmain()
{
   inta[10]= {1, 5, 8, 9, 6, 7, 3, 4, 2, 0};
   cout<< "\n The array before sorting is : ";
   show(a);

   sort(a, a+10);

   cout<< "\n\n The array after sorting is : ";
   show(a);

   return0;

}
```

The output of the above program is :
The array before sorting is : 1 5 8 9 6 7 3 4 2 0
The array after sorting is : 0 1 2 3 4 5 6 7 8 9

Searching:

It is a widely used searching algorithm that requires the array to be sorted before search is applied. The main idea behind this algorithm is to keep dividing the array in half (divide and conquer) until the element is found, or all the elements are exhausted.

It works by comparing the middle item of the array with our target, if it matches, it returns true otherwise if the middle term is greater than the target, the search is performed in the left sub-array. If the middle term is less than target, the search is performed in the right sub-array.

The prototype for binary search is :

binary_search(startaddress, endaddress, valuetofind)
startaddress: the address of the first element of the array.
endaddress: the address of the last element of the array. valuetofind: the target value which we have to search for.

//Searching

```
#include
<algorith
m>
#include
<iostream
>
Using namespace std;
voidshow(inta[], intarraysize)
{
   for(inti = 0; i
      <arraysize;
```

```
        ++i) cout<<
        a[i] << " ";
}

intmain()
{
   inta[] = { 1, 5, 8, 9, 6, 7, 3, 4, 2, 0 };
   intasize = sizeof(a) /
   sizeof(a[0]); cout<<
   "\n The array is : ";
   show(a, asize);

   cout<< "\n\nLet's say we want to search for 2 in the
   array"; cout<< "\n So, we first sort the array";
   sort(a, a + asize);
   cout<< "\n\n The array after
   sorting is : "; show(a, asize);
   cout<< "\n\nNow, we do the
   binary search";
   if(binary_search(a, a + 10, 2))
      cout<< "\nElement found
   in the array"; else
      cout<< "\nElement not found in the array";

   cout<< "\n\nNow, say we want to
   search for 10"; if(binary_search(a, a
   + 10, 10))
      cout<< "\nElement found
   in the array"; else
      cout<< "\nElement not found in the array";

   return0;
}
```

Output:

```
The array is : 1 5 8 9 0 6 7 3 4 2 0
Let's say we want to search for 2 in the array
 So, we first sort the array
The array after sorting is : 0 1 2 3 4 5 6 7 8 9
Now, we do the binary search
Element found in the array
```

Facilities:

Linux Operating Systems, G++

Algorithm:

1. Start.
2. Give a header file to use ‘vector’.
3. Create a vector naming ‘personal_records’.
4. Initialize variables to store name, birth date and telephone number.
5. Using iterator store as many records you want to store using predefined functions as push_back().
6. Create another vector ‘item_record’
7. Initialize variables to store item code, item name, quantity and cost.
8. Using iterator and predefined functions store the data.
9. Using predefined function sort(), sort the data stored according to user requirements.
10. Using predefined function search, search the element from the vector the user wants to check.

11. Display and call the functions using a menu**.**
End.

Input:

Personnel information such as name, DOB, telephone number.

Output: Display personnel information from database. The result in following format:
***** Menu *****
1.Insert
2.Display
3.Search 4.Sort
5.Delete 6.Exit
Enter your choice:1 Enter
Item Name: bat Enter Item
Quantity:2 Enter Item
Cost:50
Enter Item Code:1

Conclusion:

Hence, we have successfully studied the concept of STL(Standard Template Library) and how it makes many data structures easy. It briefs about the predefined functions of STL and their uses such a search() and sort()

Questions:

1. **What is STL?**
2. **What are four components of STL?**
3. **What is Sorting?**
4. **What is Searching?**
5. **What vector container?**